Qiong Hu (405065032)
Zihao Zou (005349580)
Gaofang Sun (104853165)

Project 3 Report
RL and IRL

ECE 232E
Large Scale Networks
May 24, 2020

2 Reinforcement learning (RL)

2.1 Environment

In the reinforcement learning part of the project, the goal is to learn the optimal policy given a single agent and the environment. The environment of the project is modeled by a Markov Decision Process (MDP), which is a tuple containing a set of states S, a set of actions A, a set of transition probabilities $P^a_{ss'}$, a set of rewards $R^a_{ss'}$, and the discount factor γ. In the process, the agent takes an action a to change its state from the current state s. By doing so, the agent acquires a new state s' and a reward R, which generally would be dependent on a, s, s' but in this project, is a function of the new state s' only.

2.1.4 Reward function

Question 1: Figure 1 shows the heat maps of two reward functions that we are going to use later, both of which are a function of the new state s'.

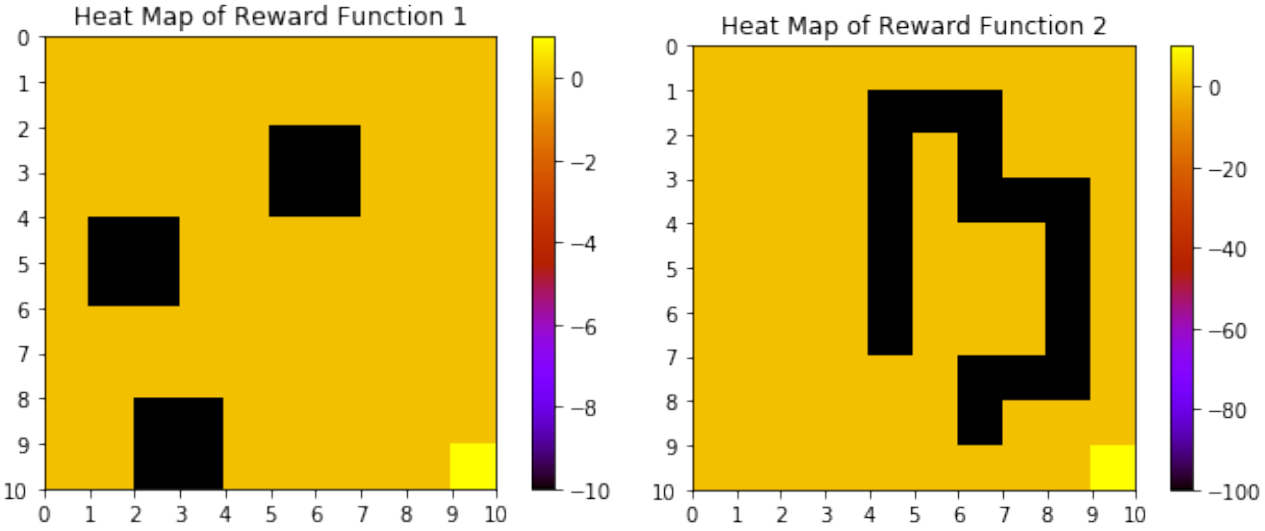


Figure 1: Heat maps for two reward functions

3 Optimal policy learning using RL algorithms

Question 2: To implement MDP, we have the set of action A that includes four actions – moving up, down, left and right, and the set of state S which is a 2-D 10×10 square grid space with 100 states. We already plot the reward functions as above. Each transition probability $P^a_{ss'}$ is dependent on the current state s, the action a, and the next state s'. The probability of wind blowing is also introduced. For each action, the probability of moving in a targeting direction is

$1-w$ and the probability of wind blowing to any random direction is $\frac{w}{4}$. If the action is to move off the grid, the agent stays at the current space with the probability of that action. The discount factor here is set to $\gamma = 0.8$.

We use the Value iteration algorithm to look for an optimal state-value function. Figure 2 shows the optimal value for Reward Function 1 when $w = 0.1$.

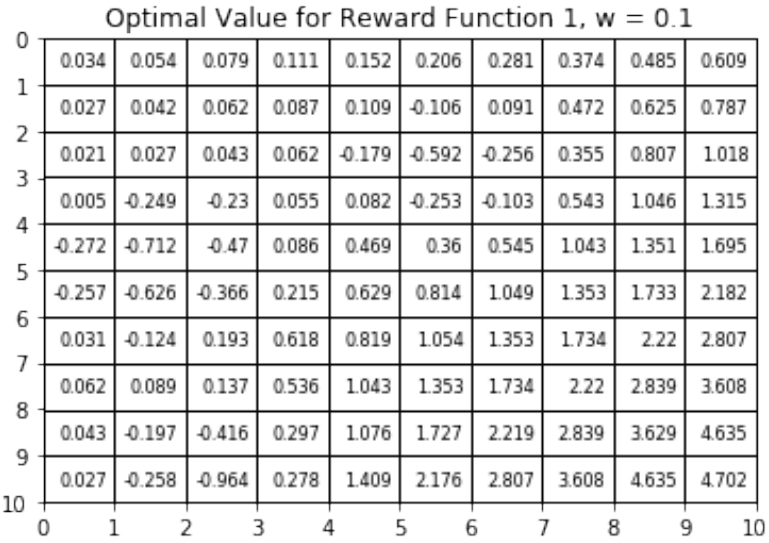
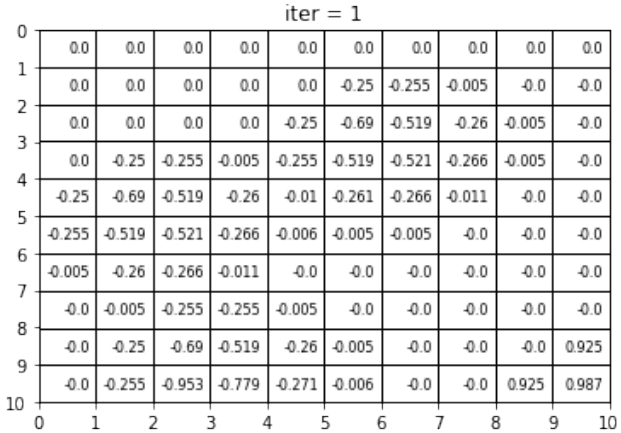
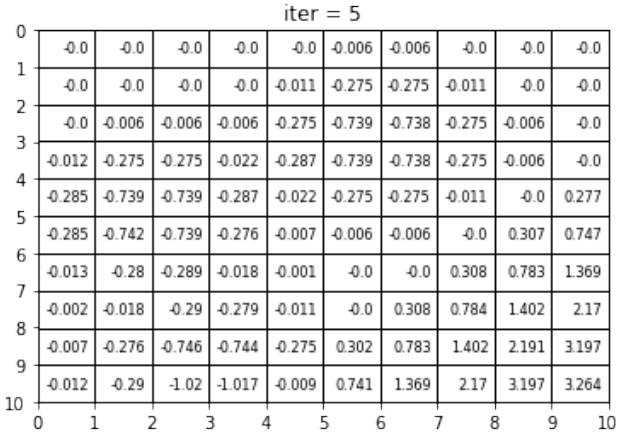


Figure 2: Optimal state values for Reward Function 1, $w = 0.1$

To converge into the optimal value set so that $\Delta \leqslant \epsilon = 0.01$, the process took 21 iterations. The following are the snapshots for the 5 different steps linearly distributed from 1 to 21, at step 1, 5, 9, 13, 17, and the final step 21.





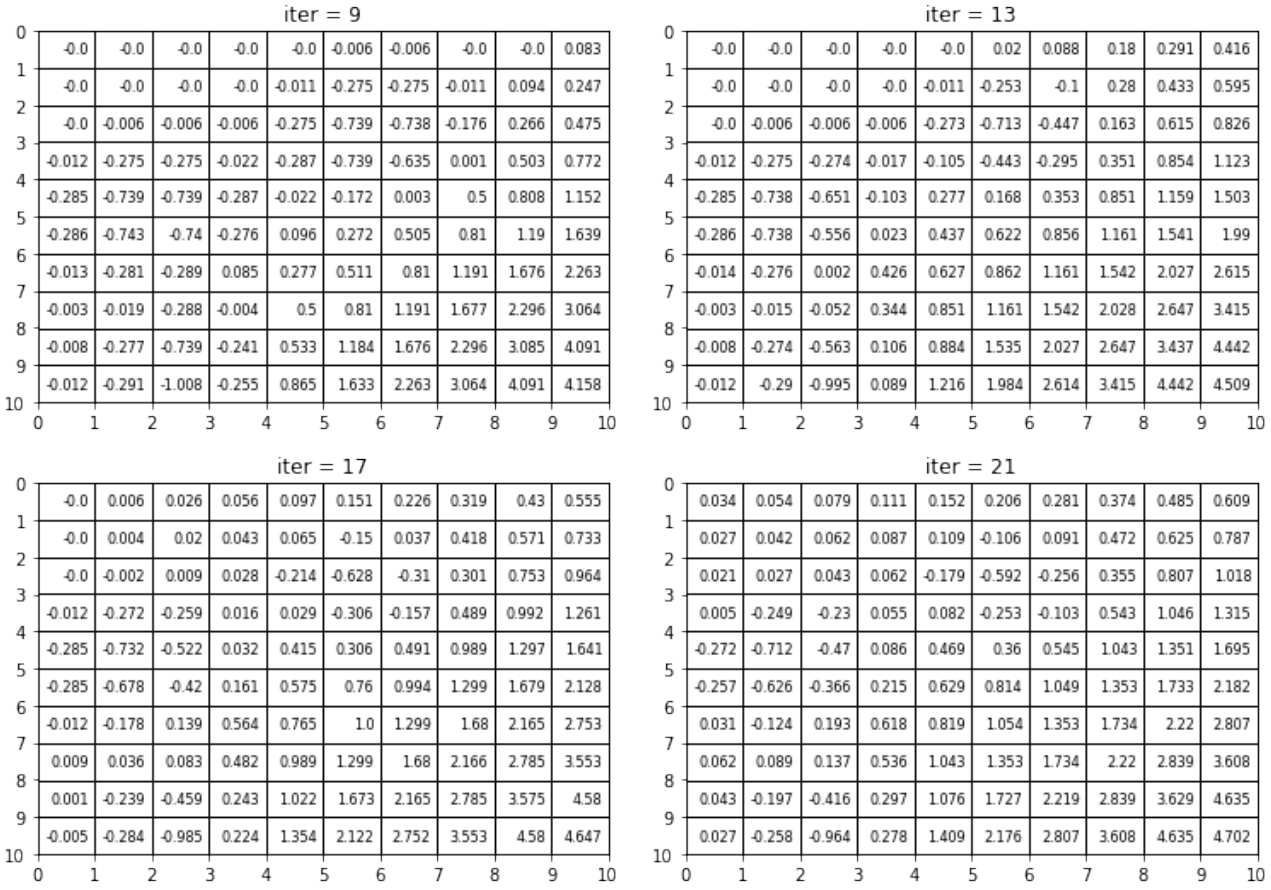


Figure 3: State values at step 1, 5, 9, 13, 17, and 21

We observed that the state values first update around the states with non-zero reward values, and then slowly spread to the whole state space. During the iteration, the state that starts with the maximum positive reward has a gradually growing value and maintains to have the largest value among all the states.

Question 3: Figure 4 is the heat map of the optimal state values from Figure 2.

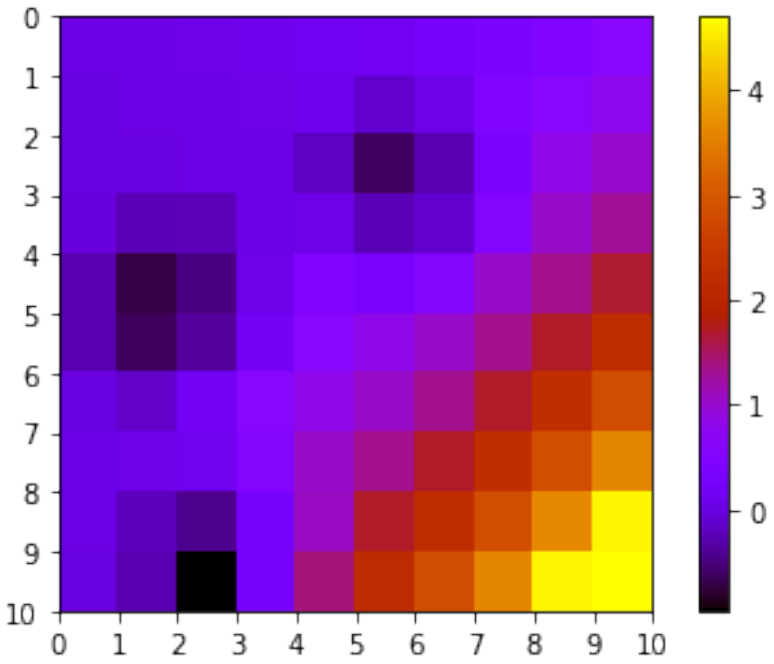


Figure 4: Heat Map of the optimal state values for Reward Function 1, $w = 0.1$

Question 4: Across the 2-D grid, Figure 4 shows the optimal state values for Reward Function 1. Since the state value depends on both its own reward and the rewards of its neighbors, theoretically the distribution should follow the rule that the area near the state with higher reward should have a higher optimal value, and the nearer the higher. From the heat map, we can see intuitively that the lower right corner, the state with index = 99, has the highest reward, and hence, it also has the highest state value after iterations. And the states near the state 99 has a gradually decreasing state value as the distance goes farther, showing a rough pattern of diagonal symmetry in color. Also, the state blocks that represent obstacles have negative minimum rewards, and thus these states and their neighbors also tend to have a low state value, and slowly increases as the distance goes farther, which is represented by a trend of decay in color from black to dark blue and then light blue.

Question 5: The optimal policy of the agent at each state is determined by the action that would return the maximum value among four actions. Figure 5 shows the optimal policy for Reward Function 1 according to the optimal values in Figure 2 from Question 2. The optimal actions are displayed using arrows.

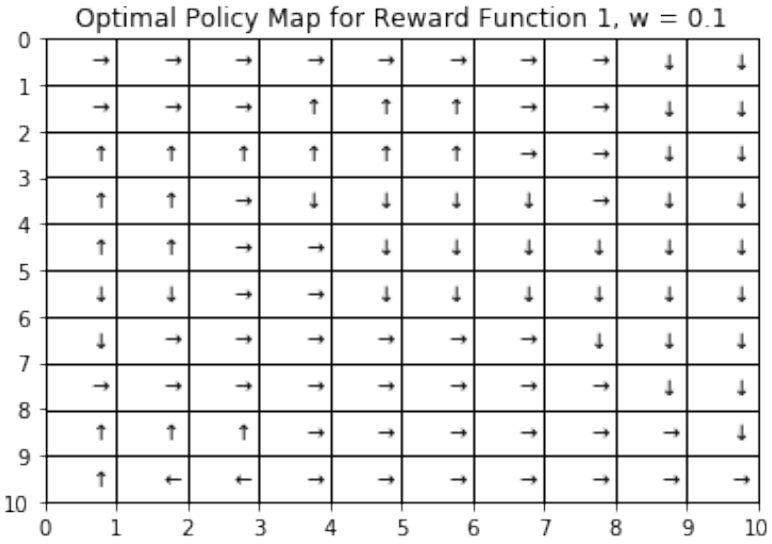


Figure 5: Optimal policy map for Reward Function 1, $w = 0.1$

This optimal policy of the agent matches our intuition. As explained in Question 4, the maximum reward and state value is obtained at the lower right corner, state with index = 99, and the states near the obstacles have smaller state values. Therefore, the action arrows show a pattern to point towards the state 99 and away from the obstacles.

From observation and comparison of Figure 5 and Figure 2, it is possible for the agent to determine the optimal policy by observing the optimal values of its neighboring states. At each state position, the agent can take the action that leads to the highest state value among its neighbors. It may not necessarily always work for other system with an optimal values that have local maximum state values, but luckily, in the situation of Reward Function 1, by taking the policy that moves towards a state with highest possible optimal value, the agent should end up in the lower right corner as expected.

Question 6: In this problem, we use the same environment and value iteration algorithm as before, except replacing Reward Function 1 with Reward Function 2. Figure 6 shows the optimal state values for Reward Function 2 after 31 iterations so that $\Delta \leqslant \epsilon = 0.01$.

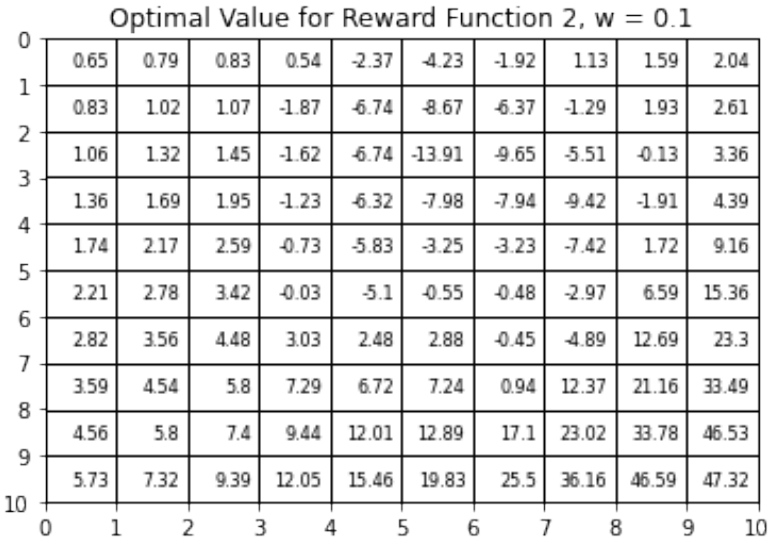


Figure 6: Optimal state values for Reward Function 2, $w = 0.1$

Similar to the optimal state values for Reward Function 1 in Figure 2, the lower right corner in Figure 6, the state with index = 99, has the maximum reward and optimal value; the states that are close to the state 99 tend to have large positive values, and the states that are close to the blocks with negative rewards also tend to have low values.

Question 7: Figure 7 is the heat map of the optimal state values from Figure 6.

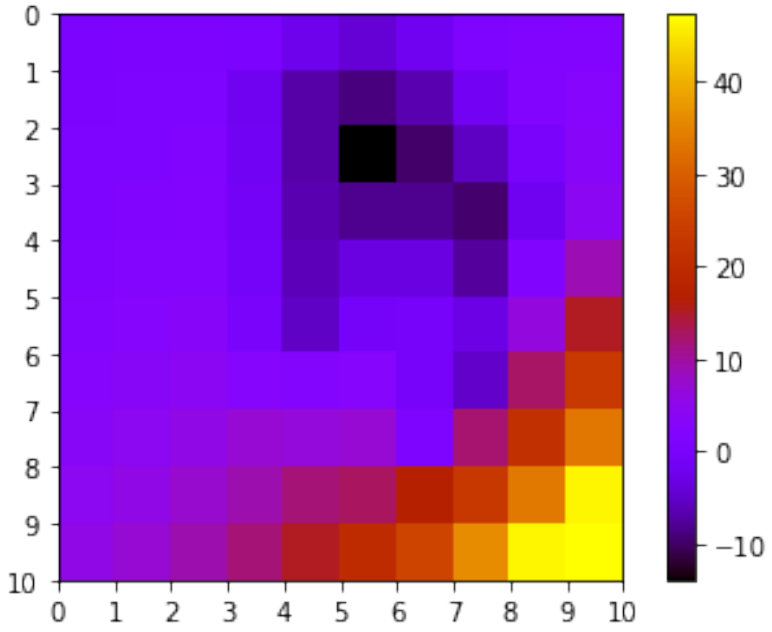


Figure 7: Heat map of optimal state values for Reward Function 2, $w = 0.1$

Near the lower right corner of the state space, the distribution in Figure 7 shows a similar pattern as in Figure 4 for Reward Function 1. The state with index = 99 has the largest reward and optimal value, thus also has the brightest color in the heat map. The states that are close to the state 99 have a gradually decreasing value number as the distance between them gets larger, and the distribution roughly shows a pattern of diagonal symmetry. The obstacle states with negative rewards, along with their neighbors, tend to have a small optimal value, which are represented by the dark colors in the heat map. The rest of the states fill in the gradual transition so that the agent would be led towards the targeting state 99 whatever state it starts with.

Question 8: Figure 8 shows the optimal policy for Reward Function 2 with $w = 0.1$. The optimal actions are displayed using arrows.

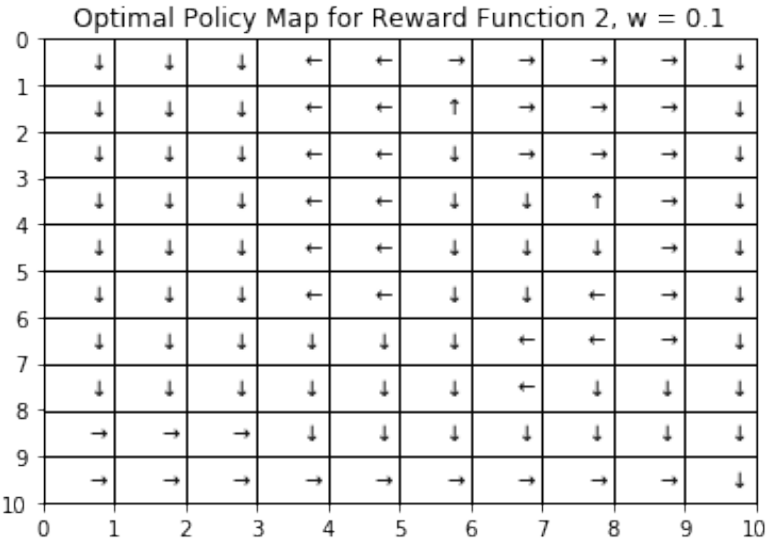


Figure 8: Optimal policy map for reward function 2, $w = 0.1$

The optimal policy map for reward function 2 also matches our intuition. As we observed from Figure 6 and Figure 7, the state with index = 53 and coordinate (2, 5) has the darkest color on the heat map, meaning it has the lowest optimal value. And thus in Figure 8, the optimal policy of the states surrounding it shows a pattern to move away from it, which makes sense. Moreover, the general policy of all the states is to move towards the state with index = 99, which is also within expectation.

Question 9: The six subfigures in Figure 9 show the optimal state values, the heat maps of the optimal state values, and the optimal policy maps for both reward functions when $w = 0.6$.

Qiong Hu (405065032)
Zihao Zou (005349580)
Gaofang Sun (104853165)

Project 3 Report
RL and IRL

ECE 232E
Large Scale Networks
May 24, 2020

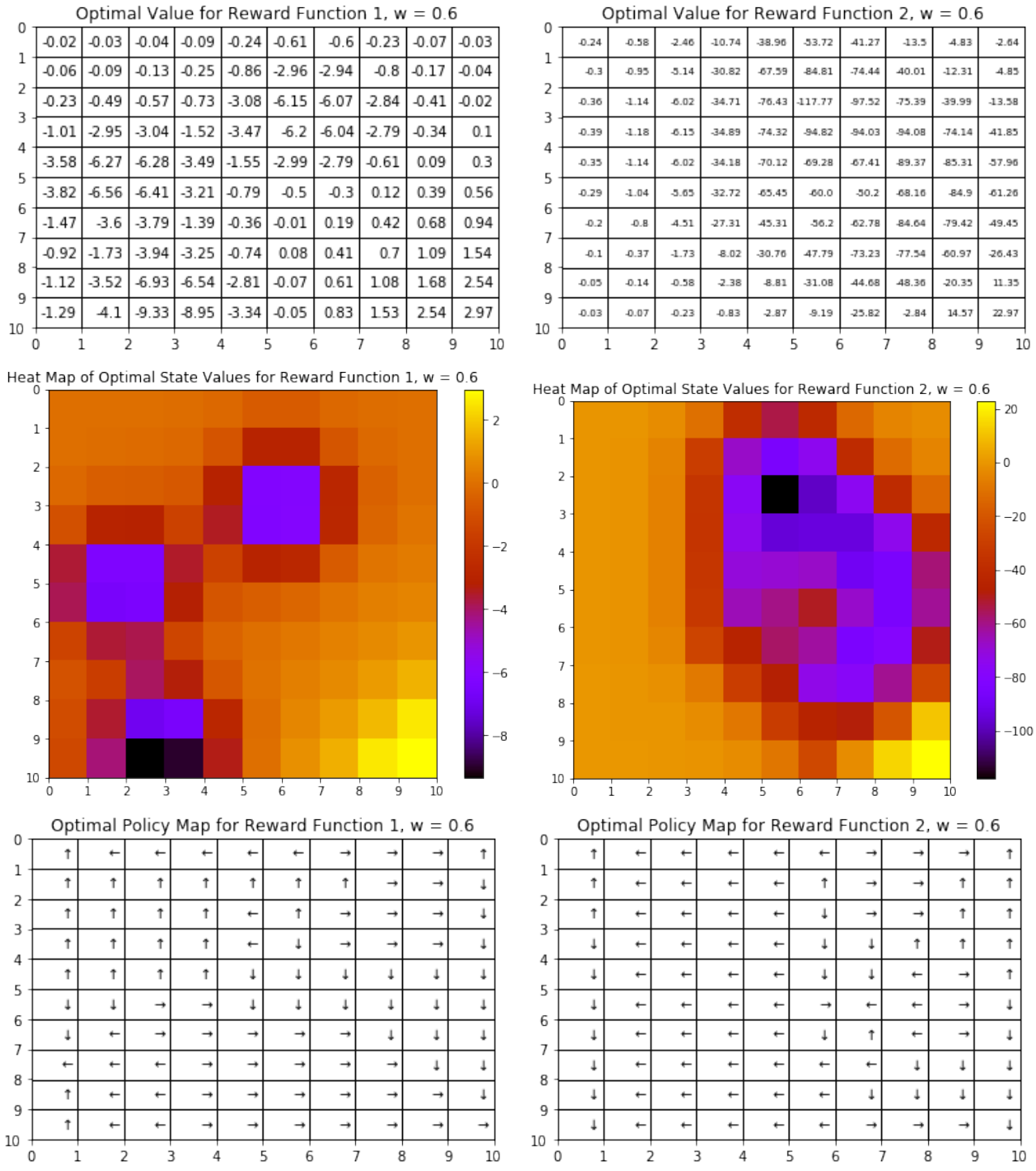


Figure 9: Optimal state values, their heat maps, and the optimal policy maps for two reward functions when $w = 0.6$

After changing the hyper parameter w from 0.1 to 0.6, we noticed many differences in the optimal value and policy for both reward functions from Figure 9.

To start with, in both optimal value tables, the state value of the maximum-reward state with index=99 has a smaller value compared to their corresponding value when $w = 0.1$. And the states that are close to the negative-reward blocks tend to have an even smaller value.

In the two heat maps of $w = 0.6$, we can see intuitively that the general color of the whole map transit from blue to orange, meaning that the general optimal state value of $w = 0.6$ is closer to the maximum value than that of $w = 0.1$. However, the blue areas near the negative-reward blocks also enlarge, showing a larger uncertainty among the whole state space.

Unlike the optimal policy maps in Figure 5 and Figure 8 when $w = 0.1$, where the policy for most states would lead towards the lower right corner state with index = 99, in the situation of $w = 0.6$, the new optimal policy maps have a higher possibility to lead the agent moving towards the four corners of the state map as a local optimal result, instead of global optimal result.

The reason of these differences can be explained intuitively by the definition of w. w is defined as the possibility that the agent takes a random action among four possible actions due to wind flow. When w changes from 0.1 to 0.6, the agent has a higher possibility to take random actions off the targeting direction, leading to a larger noise added to the state value, and thus the change in w also makes the optimal policy more ambiguous.

It is also explainable why the policy map would lead the agent to four corners as local optimal results instead of the lower right corner with the maximum reward and value when $w = 0.6$. When the agent is far away from the lower right corner and closer to the negative-reward blocks, it is easier for the agent to be flown away from the blocks instead of moving towards the target, which explains the behavior of the generated optimal policy at $w = 0.6$.

According to the analysis and explains above, we think $w = 0.1$ would give rise to better optimal policy, and therefore we are going to use $w = 0.1$ for the next stage of the project.

4 Inverse Reinforcement learning (IRL)

4.1 IRL algorithm

In this part of the project, we will use IRL algorithm to extract the reward function from the optimal policy computed in the previous section as the expert behavior. Then, we will use the extracted reward function to generate a new optimal policy of the agent, and compare the two policies using some similarity metric to measure the performance of the IRL algorithm.

Question 10: Based on the IRL algorithm, we can convert the regular Linear Programming (LP) formulation into the an equivalent form with block matrices, as expressed below.

$$\mathbf{c} = \begin{pmatrix} \mathbf{I} \\ -\lambda \cdot \mathbf{I} \\ \mathbf{0} \end{pmatrix} \in \mathbb{R}^{3|\mathcal{S}| \times |\mathcal{S}|}, \tag{1}$$

$$\mathbf{x} = \begin{pmatrix} \mathbf{t} \\ \mathbf{u} \\ \mathbf{R} \end{pmatrix} \in \mathbb{R}^{3|\mathcal{S}|\times 1}, \text{where } \mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_{|\mathcal{S}|} \end{pmatrix}, \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{|\mathcal{S}|} \end{pmatrix}, \mathbf{R}_i = \mathbf{R}(s_i), \tag{2}$$

$$\mathbf{D} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & -(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma \mathbf{P}_{a_1})^{-1} \\ \mathbf{I} & \mathbf{0} & -(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma \mathbf{P}_{a_1})^{-1} \\ \mathbf{0} & -\mathbf{I} & -\mathbf{I} \\ \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & -\mathbf{I} \end{pmatrix} \in \mathbb{R}^{6|\mathcal{S}|\times 3|\mathcal{S}|}, \forall a \in \mathcal{A} \backslash a_1, \tag{3}$$

$$\mathbf{b} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ R_{max} \cdot \mathbb{1} \\ R_{max} \cdot \mathbb{1} \end{pmatrix} \in \mathbb{R}^{6|\mathcal{S}|\times 1}. \tag{4}$$

4.2 Performance measure

Question 11: We sweep λ through 0 to 5 to get 500 evenly spaced values to look for the λ that has the best performance of the IRL algorithm. To evaluate the performance, we define accuracy as the ratio of states that have the same action as the ground truth in Figure 5 from Question 5. The relationship between the accuracy and the parameter λ is plotted in Figure 10.

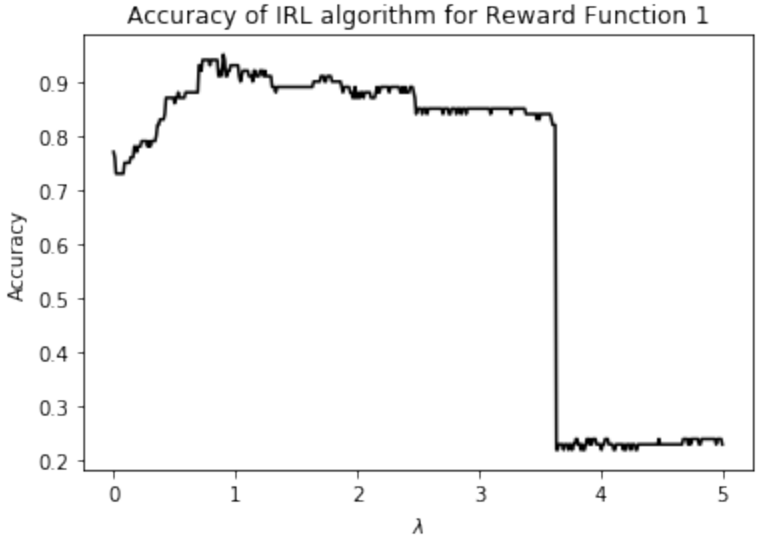


Figure 10: Accuracy of the IRL algorithm for Reward Function 1

From Figure 10 we noticed that the accuracy of the IRL algorithm increases as λ increases when $\lambda < 1$ and remain almost stable until $\lambda = 3.63$, and when $\lambda > 3.63$ the accuracy shoots straight down from about 0.83 to remain around 0.22.

Question 12: From the calculation for Figure 10 in Question 11, we find that when $\lambda = 0.902$, the accuracy reaches the maximum value 0.95. Therefore, we will use the value $\lambda_{max}^{(1)} = 0.902$ as the λ for extracting Reward Function 1 for future reference.

Question 13: We use $\lambda_{max}^{(1)}$ in Question 12 to generate the heat map of the extracted reward function and compare it to the ground truth Reward Function 1 in Figure 1 from Question 1.

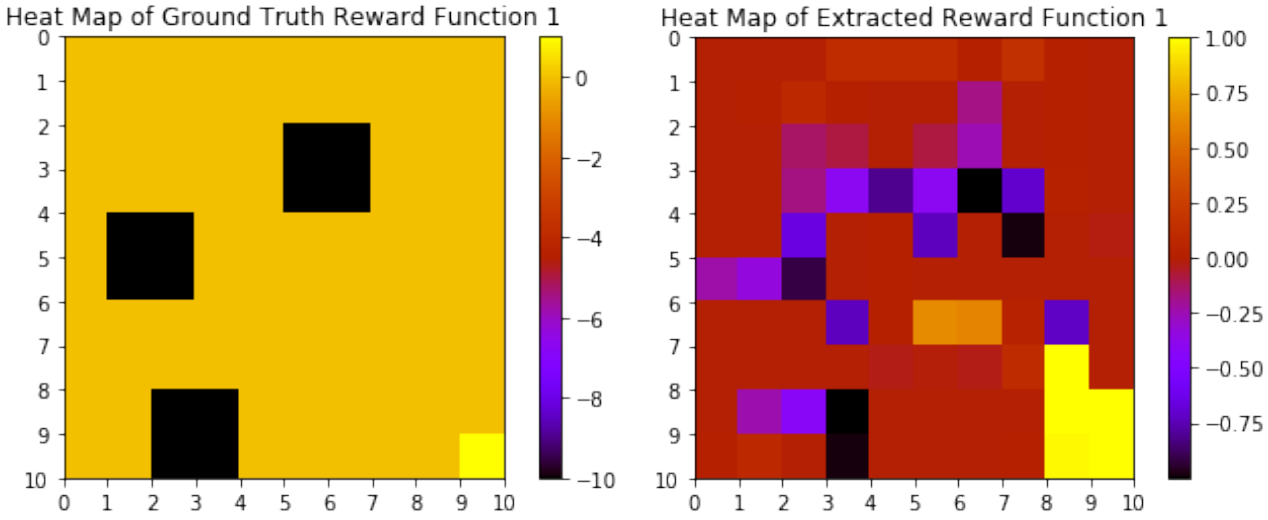


Figure 11: Heat maps of ground truth and extracted Reward Function 1

Comparing the two heat maps in Figure 11, we noticed that the IRL algorithm successfully extract the maximum reward state with the index = 99, and correctly estimate the areas in or near the negative-reward blocks in the ground truth, even though there are quite some noises.

Question 14: We use the extracted reward function computed in Question 13, $\lambda = \lambda_{max}^{(1)}$ and $w = 0.1$ to compute the new optimal state values in the 2-D grid state space. For computing the optimal values, we use the same optimal state-value function from Question 2. The optimal state-value reaches $\epsilon = 0.01$ after 18 iterations. The resulting heat map is shown in Figure 12.

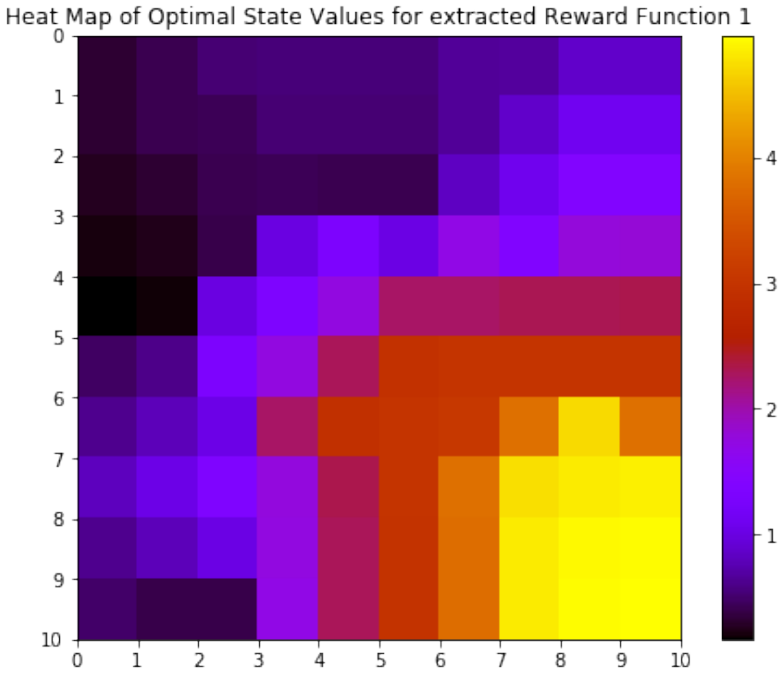


Figure 12: Heat map of optimal state values for extracted Reward Function 1, $w = 0.1$

Question 15: By comparing the heat maps in Figure 4 and Figure 12, we notice a similar trend in the overall state values, where the maximum state value appears in the lower right corner with state index = 99, and decreases along the direction of diagonal towards the upper left corner. The reason is that the extracted reward function in Figure 12 correctly extract the feature that the state 99 has the maximum reward and thus leading to a maximum state value. And in the state areas where the influence of the negative-reward blocks are not significant, the state values from both ground truth and extracted reward functions are similar during the process of iteration.

The difference mainly appear in the areas in or near the negative-reward states, where in Figure 4, the state values reflect a clear pattern of those negative-reward blocks, while in Figure 12, the pattern is blurred and most of the upper left areas appear to have dark colors, reflecting low state values. The reason is that, near those areas, the expert policy has some uncertainty and ambiguity whether the action is due to moving away from the blocks or moving towards the targeting positive-reward state, and therefore during the processing of extraction, these areas tend to have large noises, and show more differences compared to the ground truth.

Question 16: We compute the optimal policy of the agent based on the extracted reward function in Question 13 using the same algorithm as in Question 5. The optimal policy map is shown in Figure 13, using arrows to represent the action in each state.

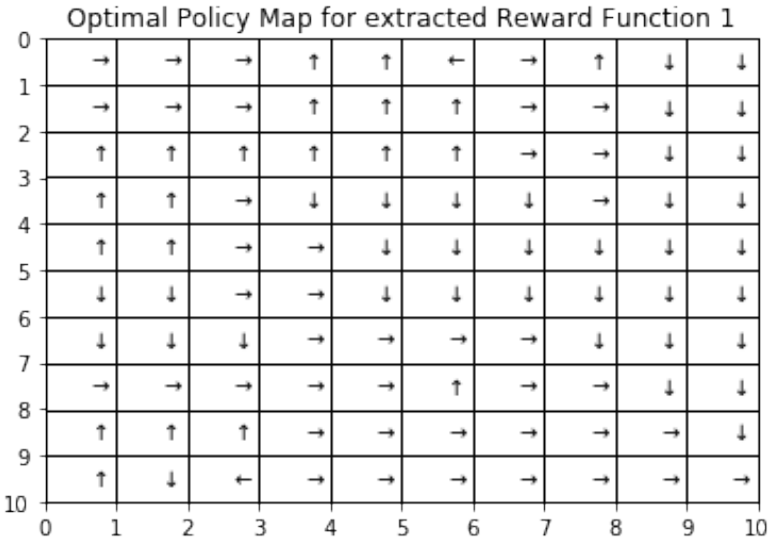


Figure 13: Optimal policy map for extracted Reward Function 1, $w = 0.1$

Question 17: Comparing the policy map in Figure 5 of Question 5 and Figure 13, we plot the comparison result in Figure 14. For each state, if the policy is the same in two situations, a single policy is plotted, otherwise both policies are plotted with the ground truth on the left and the policy from the extracted reward function on the right of the backslash sign.

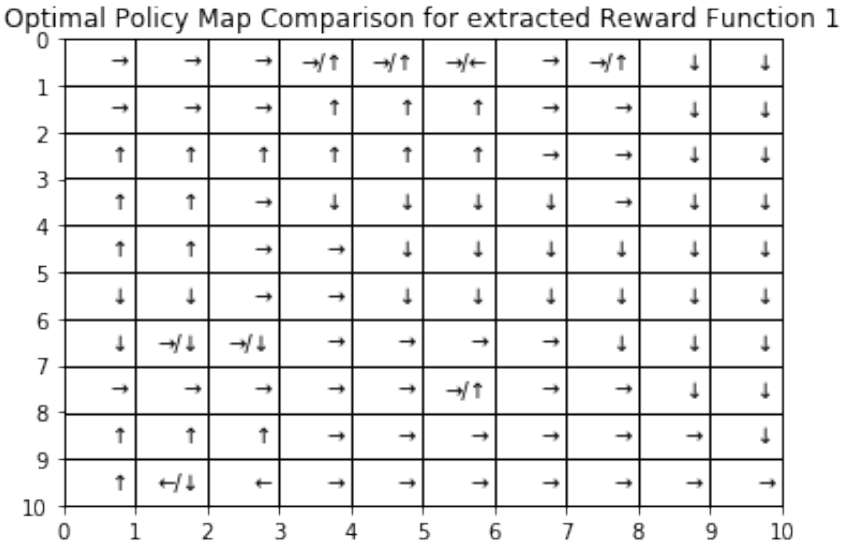


Figure 14: Optimal policy comparison between Figure 5 and Figure 13

From Figure 14, we find that most of the states have the same policy for action. Both of them follow the rule to move away from the negative-reward blocks and towards the lower right corner with state index = 99. The reason is that the extracted reward function shares the same general

feature of the reward distribution, where the location of the maximum reward is correctly extracted and the areas of the negative-reward is roughly estimated.

However, we also notice some minor differences. In Figure 13, the states with coordinate (0,3), (0,4), (0,7) and (9,1) choose the policy to move off the grid which means they would have a high probability to stay in the current state during iteration, while in Figure 5, their optimal policy decision would lead them towards their neighboring states. A possible reason might be that, they are close to or surrounded by the estimated negative-reward states, leading to a local maximum value for the state, or the threshold of the iteration $\epsilon = 0.01$ is too large for these states to find a better policy. The problem might be eliminated if ϵ is set to be a smaller number.

Question 18: We sweep λ from 0 to 5 to get 500 evenly spaced values to look for the λ that has the best performance for IRL algorithm. We use the optimal policy in Figure 8 of Question 8 as the expert policy, and the same method to calculate the accuracy. The plot of Accuracy against λ is shown in Figure 15.

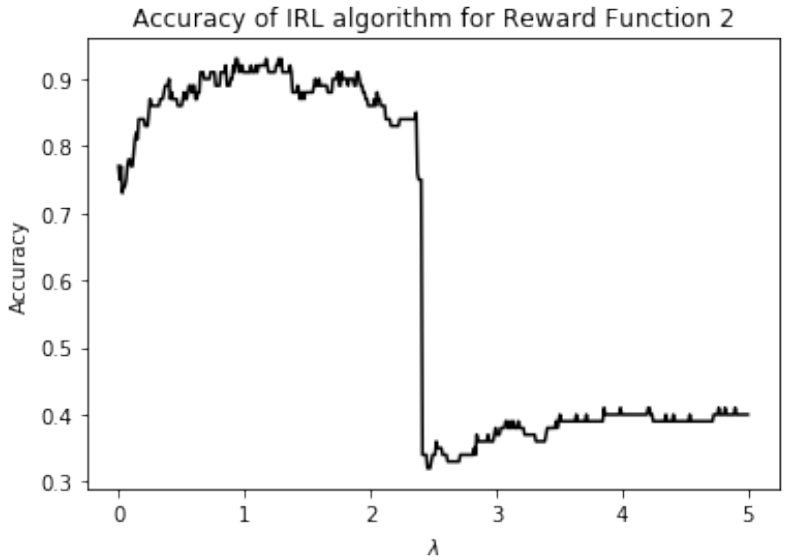


Figure 15: Accuracy of the IRL algorithm for Reward Function 2

We find a similarity in the curve trend between Figure 10 and Figure 15. Both curves have a rising accuracy as λ increases, remain stable until λ reaches a cut-off point, and then drop down rapidly and remain low. The cut-off point of λ for Reward Function 1 is around 3.63, and for Reward Function 2 is around 2.40.

Question 19: In Figure 15, when $\lambda = 0.932$, the accuracy reaches the maximum value 0.93. Therefore, we will use the value $\lambda^{(2)}_{max} = 0.932$ for extracting Reward Function 2 for future reference.

Question 20: We generate heat maps of the ground truth and extracted Reward Function 2 as shown in Figure 16, using $\lambda = \lambda^{(2)}_{max}$.

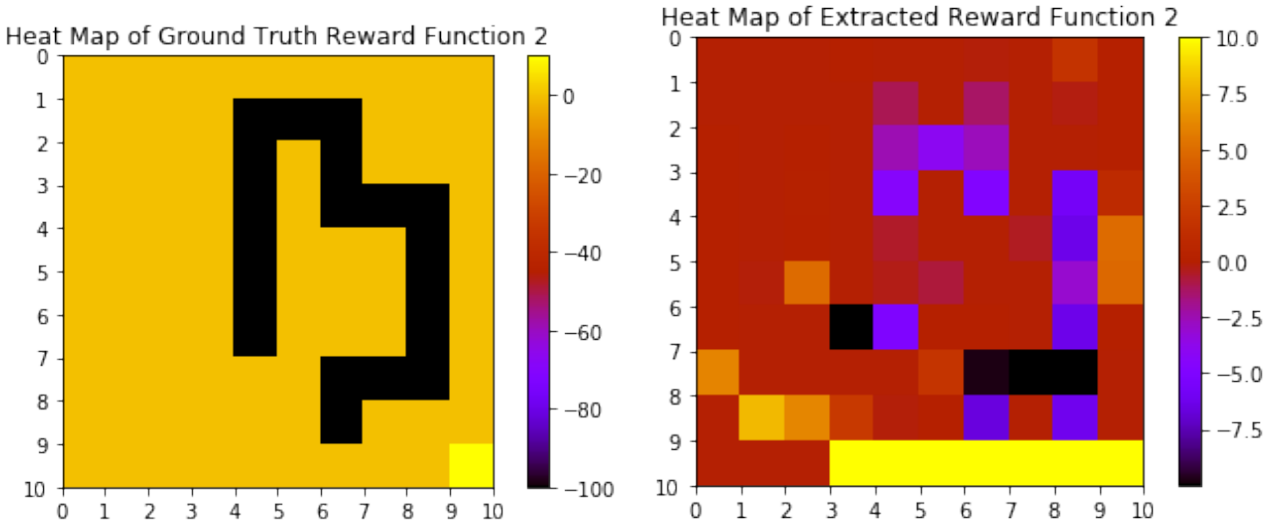


Figure 16: Heat maps of ground truth and extracted Reward Function 2

We notice that the difference between the ground truth Reward Function 2 and its corresponding extracted reward function in Figure 16 is more obvious than that of Reward Function 1 in Figure 11. The inversed reward function roughly extract the feature that the lower right corner has high reward, but somehow, it marks almost the whole bottom line as a high reward. It also roughly extract the outline of the original negative-reward obstacles with much noises. The rest of the state space is estimated to be zero-reward which is correct.

Question 21: We use the extracted reward function in Question 20, $\lambda = \lambda_{max}^{(2)}$, and $w = 0.1$ to compute the new optimal state value in the 2-D grid state space. For computing the optimal values, we use the same optimal state-value function from Question 2. The optimal state-value reaches $\epsilon = 0.01$ after 25 iterations. The resulting heat map is shown in Figure 17.

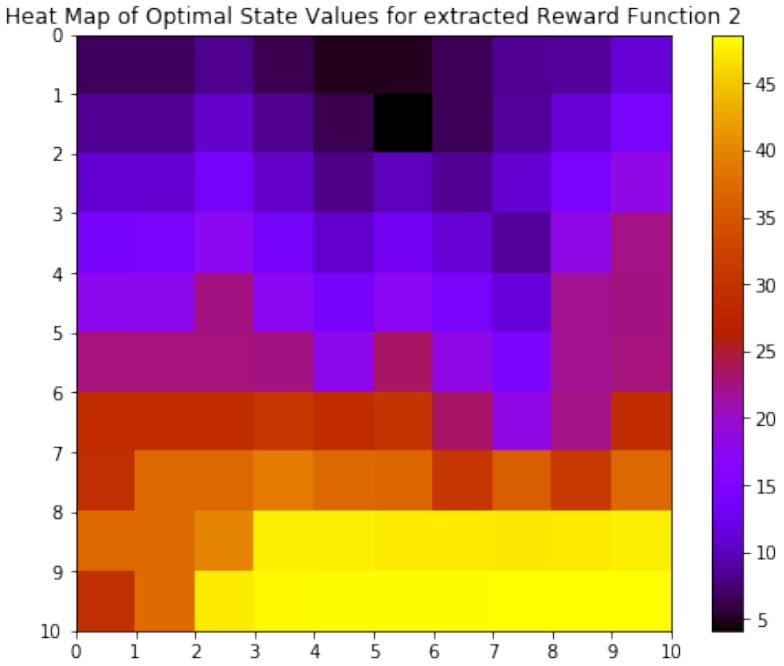


Figure 17: Heat map of optimal state value for extracted Reward Function 2, $w = 0.1$

Question 22: By comparing the heat maps of optimal state-value in Figure 7 of Question 7 and Figure 17, we notice more differences than similarities. A rough similarity is that, in both figures, the state with index=99 has the maximum state value, and the minimum state value falls into the state somewhere near the upper part of the state space. Figure 7 has the minimum state value at the state with coordination (2, 5) and in Figure 16 the coordinate is (1, 5).

However, the differences are more obvious. Due to the noisy estimation of the reward function, where most of the lower part of the state space have high reward, they also show large state values. The general pattern in Figure 16 shows a rough symmetry in x-axis while in Figure 7 it's more like diagonal symmetry. Additionally, Figure 7 roughly reflects the pattern of continuous obstacle states, but it is almost completely blurred in Figure 16. It is easy to understand since the extracted reward function is already noisy, the process of state-value calculation adds more noise due to wind blowing, therefore the state-value plot no longer shows the outline of the snake-look obstacles.

Question 23: We compute the optimal policy of the agent based on the extracted reward function in Question 20 using the same algorithm as in Question 8. The optimal policy map is shown in Figure 18, using arrows to represent the action in each state.

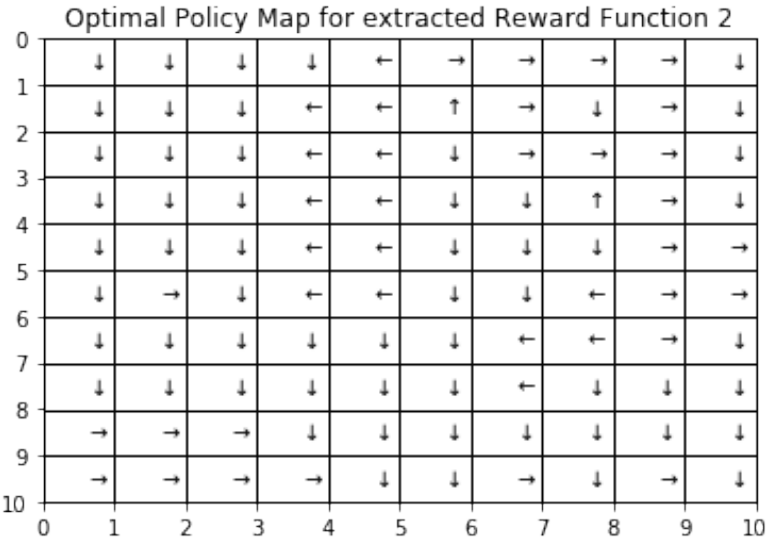


Figure 18: Optimal policy map for extracted Reward Function 2, $w = 0.1$

Question 24: The question asks about the comparison of the Figure 9 of Question 9 and Figure 18, but since we decide to use $w = 0.1$ for Figure 18 and $w = 0.6$ for Figure 9, there is no point to compare these two results. Therefore, we are going to compare Figure 18 with Figure 8 of Question 8, where both figures show the optimal policy map at $w = 0.1$.

Similar to Question 17, we generate a comparison table to show the difference between the two optimal policies. For each state, if the policy is the same then a single policy is plotted in the table, otherwise both policies are plotted with the ground truth on the left and the policy from the extracted reward function on the right of the backslash sign.

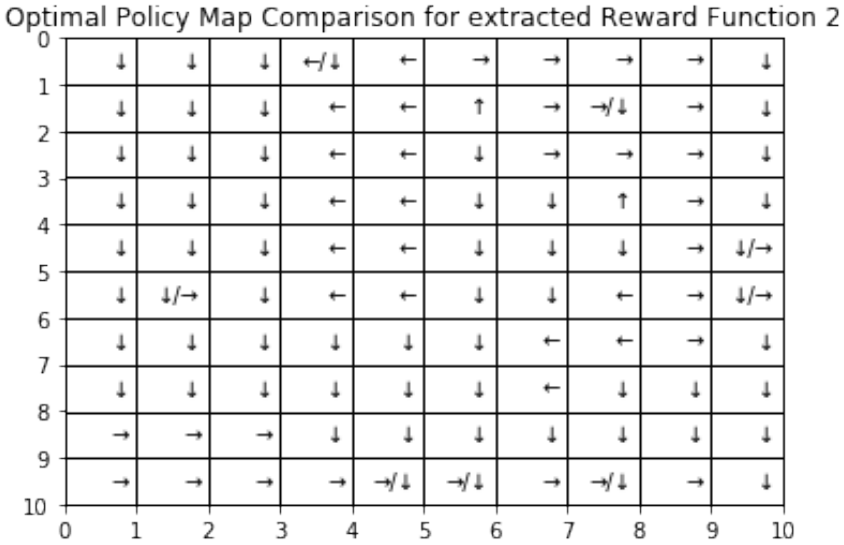


Figure 19: Optimal policy comparison between Figure 8 and Figure 18

From the comparison table in Figure 19, we notice that for most states, the optimal policy remains the same, so that the agent would move away from the estimated negative-reward obstacles and towards the state with index = 99.

Among the few states that show different policies, we notice that the states with coordinate (9,4), (9,5), (9,7), (4,9), (5,9) share something in common. These states are all on the edge of the grid state space, and their optimal policies from the extracted reward function take the action that the agent would try to move off the grid, and end up with a high probability to remain in the same old state. A possible reason is that under the current threshold $\epsilon = 0.01$, their state values become local maximum, thus the optimal action is to remain in the current state. The potential solution for the problem is to reduce ϵ to a small number.

Question 25: One of the major discrepancies have been explained in Question 24, which is the ϵ being too high, causing the policy not having enough iterations to look a the better solution. We try to fix the problem by modifying ϵ from 0.01 to $1e^{-7}$ and recompute the accuracy over different λ, as shown in Figure 20.

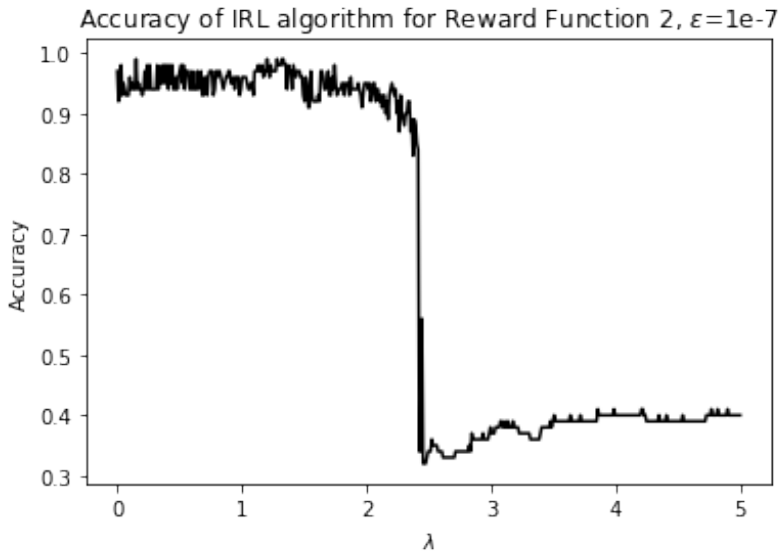


Figure 20: Accuracy of the IRL algorithm for Reward Function 2, $\epsilon = 1e^{-7}$

The trend of the accuracy curve in Figure 20 remains similar to Figure 15, except the cut-off edge is at around $\lambda = 2.42$. The accuracy reaches the maximum 0.99 at $\lambda = 0.150$. Therefore, there is a change of rising in the maximum accuracy when ϵ decreases.

By using the new $\lambda_{max} = 0.150$, $\epsilon = 1e^{-7}$, $w = 0.1$, we re-estimate the extracted reward function and state values, as shown in Figure 21. The new state values are calculated after 70 iterations.

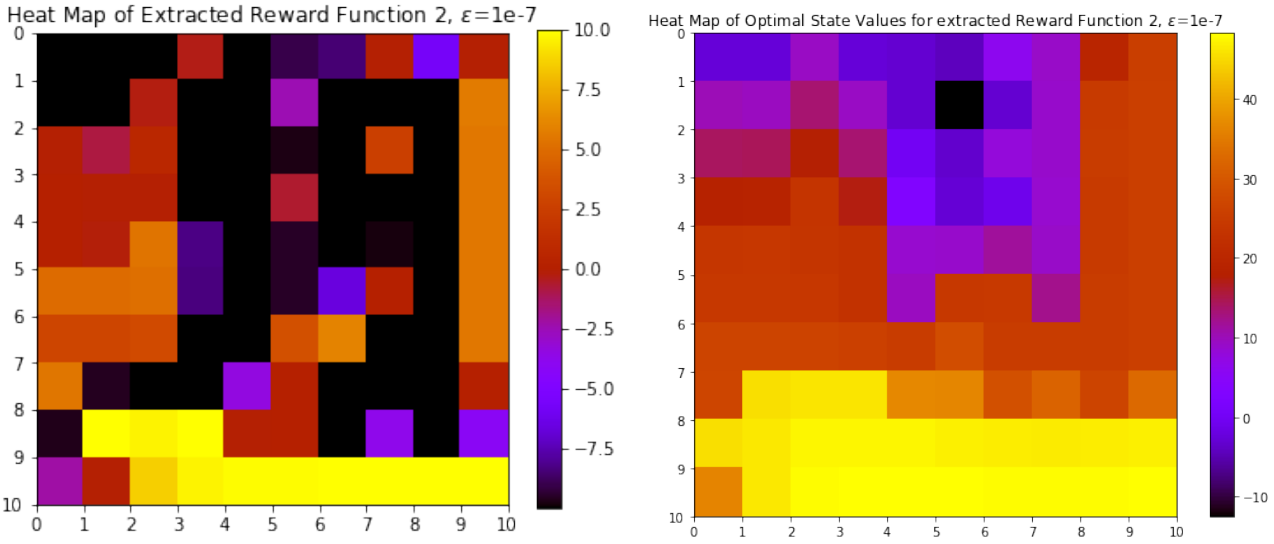


Figure 21: Heat map of extracted reward function 2 and new optimal values, $w = 0.1$, $\epsilon = 1e^{-7}$

The new estimated reward function shows a clearer pattern of the original negative-reward obstacle states. The new optimal values looks similar to Figure 17 when $\epsilon = 0.01$, but the colorbar shows that the value number are lower than Figure 17 and get closer numbers to the ground truth in Figure 7.

From the reward function and optimal state values in Figure 21, we recalculate the optimal policy and compare to the ground truth policy in Figure 8 of Question 8. The comparison result is shown in Figure 22.

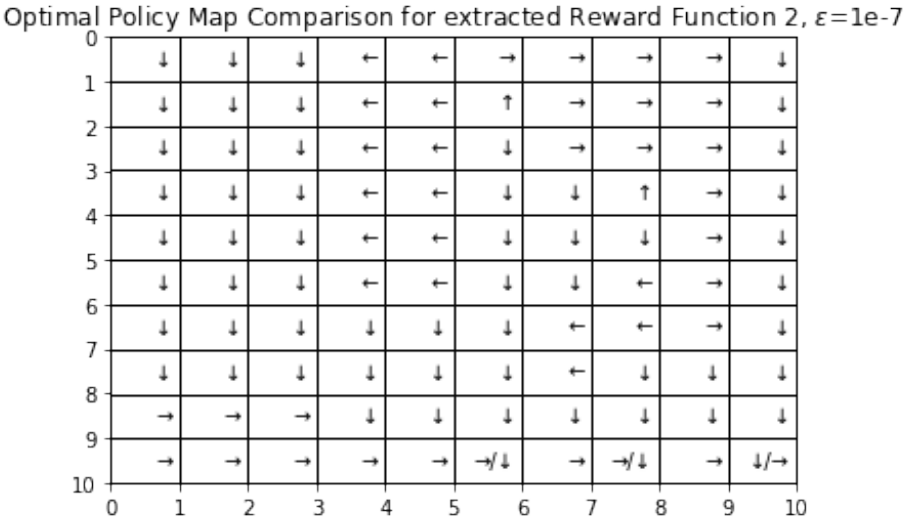


Figure 22: Optimal policy comparison to Figure 8

From Figure 22, we notice that there are only three states having different optimal policies now.

Most of the edge states that have a policy to move off the grid in Figure 18 of Question 23 now takes the action towards the state 99, showing that the first major discrepancies has been almost fixed.

However, the remaining three states reflect the second discrepancy, that the IRL algorithm has some difficulty to extract the reward of the bottom line of the state space near the maximum-reward state, and the new policy is hard to achieve 100% accuracy even if ϵ is already very low. The reason probably is that IRL is an underspecified problem and the simple IRL algorithm uses linear programming for approximation. As long as the observation of the expert policy is not infinite, it is difficult to extract the reward function very accurately, leading to some inaccuracy in the extracted optimal policy.